

Supplementary Software 2

Tree-finding algorithm

Initialise:

$s \leftarrow (1, 2, \dots, k)$ // s is a list of nodes (subclones).

procedure FINDROOTS(s) // Fill node list r .

Identify all nodes in s which could be the root node of the tree;

Place these candidate root nodes in a new list r ;

end procedure

procedure FILLTREELIST(s, r) // Fill tree list T .

$T \leftarrow ()$; // T is an empty tree list.

for each node a in r **do**

$o \leftarrow (0, 0, \dots, 0)$; // $o = (o[1], o[2], \dots, o[k])$ is a tree
// in which no node has yet
// been assigned a parent.

$U \leftarrow (o)$; // U is a tree list

for each node i in s (excluding node a) **do**

$V \leftarrow ()$; // V is an empty tree list

if U is NOT empty **then**

for each tree t in U **do** // $t = (t[1], t[2], \dots, t[k])$ is a tree.

Find every possible parent node of node i in tree t ;

Save all these possible parent nodes in a list $p(i, t)$;

for each node b in $p(i, t)$ **do**

$t' \leftarrow t$;

$t'[i] \leftarrow b$; // specifies that node b is the parent
// of node i .

Insert tree t' into list V ;

end for

end for

Delete element t from list U ;

Insert elements of list V into list U ;

end if

end for

Insert elements of list U into list T ;

end for

end procedure
